Computational social science

Counting things & analysing text

Data & Digital Traces

Thanks for the feedback!

Nonreactivity of big data

- People can change their behaviour when they know they are being observed by researchers (reactivity)
- Collected in the background, without people paying attention to the collection

-> The study doesn't change the behaviour

-> Might not be "unfiltered" behaviour, but not because of attention of researchers (social desirability bias)

Example 1

Mobility data collected by phone operators:
People are tracked by operators without them realising it (data is aggregated)

Example 2

Social media data:
Data on discussions on Facebook don't represent private discussions, awareness that they can be seen by others

Big data for CSS research

- Social sciences comprises a lot of different disciplines
- Found/observational data was also utilised in the social sciences
- Big data is typically found data
- Comparison with questionnaires, surveys, experiments (designed data) to tease out the relative benefits and weaknesses of using big data (found data) for quantitative social science research
- No data is without its flaws, more important to reason about which types of research questions are better answered by which type of data

Course structure

Period IV

Week	Lecture	Exer. dl	Ext. dl	Topic
1	~~Feb 27~~	~~Mar 3~~	Mar 15	~~Introduction to CSS~~
2	Mar 6	Mar 10	Mar 22	Artificial societies & agent-based models
3	Mar 13	Mar 17	Mar 29	Data & digital traces
4	Mar 20	Mar 24	Apr 5	Counting things & analysing text
5	Mar 27	Mar 31	Apr 12	Social networks: structure
6	Apr 3	*	-	Introduction to the project

***Project deadline: May 26**
Project peer review: June 2

Period V

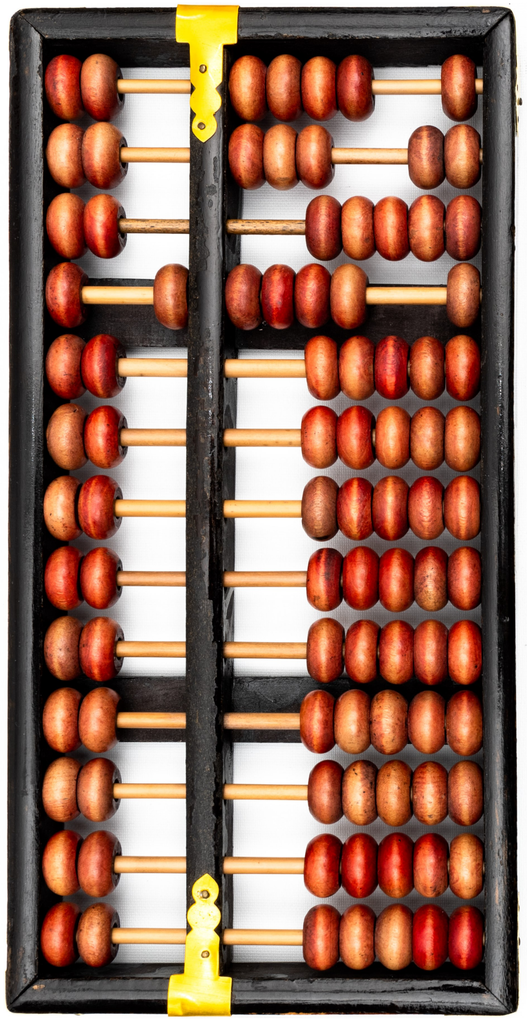
Week	Lecture	Exercise dl	Ext. dl	Topic
7	Apr 24	May 5	May 10	Ethics, privacy, legal
-	-	-	-	WAPPU
8	May 8	May 12**	May 24	Agent-based models & emergence
9	May 15	May 19***	May 31	Social networks: dynamics
10	May 22	May 26***	June 7	Experiments & interventions at scale
11	May 29	-	-	Computing for social good

****Bonus round**
*****Only lecture questions**

Counting things

- Lots of CSS research is really just counting things

- Simple counting can be interesting if you combine a good question with good data.



Felix Winkelnkemper, CC BY-SA 4.0, via Wikimedia Commons

Counting things - an example

- Data on every taxi trip taken by New York City cabs from 2009 to 2013: start time, start location, end time, end location, fare, and tip (hourly wage fluctuates)

Mariordo (Mario Roberto Durán Ortiz), CC BY-SA 3.0 via Wikimedia Commons

Farber, Henry S. 2015. “Why You Can’t Find a Taxi in the Rain and Other Labor Supply Lessons from Cab Drivers.” *Quarterly Journal of Economics* 130 (4):1975–2026. https://doi.org/10.1093/qje/qjv026.

Counting things - an example

- Data on every taxi trip taken by New York City cabs from 2009 to 2013: start time, start location, end time, end location, fare, and tip (hourly wage fluctuates)

- Goal: test two competing theories in labor economics
  - Neoclassical models predict that taxi drivers will work more on days where they have higher hourly wages
  - Behavioral economic models predict exactly the opposite

Mariordo (Mario Roberto Durán Ortiz), CC BY-SA 3.0 via Wikimedia Commons

Farber, Henry S. 2015. “Why You Can’t Find a Taxi in the Rain and Other Labor Supply Lessons from Cab Drivers.” *Quarterly Journal of Economics* 130 (4):1975–2026. https://doi.org/10.1093/qje/qjv026.

Counting things - an example

- Data on every taxi trip taken by New York City cabs from 2009 to 2013: start time, start location, end time, end location, fare, and tip (hourly wage fluctuates)

- Goal: test two competing theories in labor economics
  - Neoclassical models predict that taxi drivers will work more on days where they have higher hourly wages
  - Behavioral economic models predict exactly the opposite

- Do drivers work more hours on days with higher hourly wages (neoclassical models) or lower hourly wages (behavioral economic models)?

Mariordo (Mario Roberto Durán Ortiz), CC BY-SA 3.0 via Wikimedia Commons

Farber, Henry S. 2015. “Why You Can’t Find a Taxi in the Rain and Other Labor Supply Lessons from Cab Drivers.” *Quarterly Journal of Economics* 130 (4):1975–2026. https://doi.org/10.1093/qje/qjv026.

Counting things - an example

- Data on every taxi trip taken by New York City cabs from 2009 to 2013: start time, start location, end time, end location, fare, and tip (hourly wage fluctuates)

- Goal: test two competing theories in labor economics
  - Neoclassical models predict that taxi drivers will work more on days where they have higher hourly wages
  - Behavioral economic models predict exactly the opposite

- Do drivers work more hours on days with higher hourly wages (neoclassical models) or lower hourly wages (behavioral economic models)?

Mariordo (Mario Roberto Durán Ortiz), CC BY-SA 3.0 via Wikimedia Commons

Farber, Henry S. 2015. “Why You Can’t Find a Taxi in the Rain and Other Labor Supply Lessons from Cab Drivers.” *Quarterly Journal of Economics* 130 (4):1975–2026. https://doi.org/10.1093/qje/qjv026.

Counting things - another example

- Online censorship by the Chinese government
- Conflicting expectations about which kinds of posts are most likely to get deleted — critical of the state or posts that encourage collective behavior
- Crawled >1000 Chinese social media sites, finding relevant posts, revisiting to see if the posts were deleted
- 11 million posts on 85 different prespecified topics, each with an assumed level of sensitivity, 2 million censored
- Posts on highly sensitive topics censored only slightly more than posts on middle- and low-sensitivity topics

King, Gary, Jennifer Pan, and Margaret E. Roberts. 2013. “How Censorship in China Allows Government Criticism but Silences Collective Expression.” *American Political Science Review* 107 (2):326–43. https://doi.org/10.1017/S0003055413000014.

Counting things - another example

- The censorship rate by the topic could be misleading
- Classify posts as critical of the state, supportive of the state, or irrelevant or factual reports about the events
- Three types of posts were regularly censored: pornography, criticism of censors, and those that had collective action potential (i.e., the possibility of leading to large-scale protests)
- Could learn how the censors work just by watching and counting

King, Gary, Jennifer Pan, and Margaret E. Roberts. 2013. “How Censorship in China Allows Government Criticism but Silences Collective Expression.” *American Political Science Review* 107 (2):326–43. https://doi.org/10.1017/S0003055413000014.

Counting things - another example

- The censorship rate by the topic could be misleading
- Classify posts as critical of the state, supportive of the state, or irrelevant or factual reports about the events
- Three types of posts were regularly censored: pornography, criticism of censors, and those that had collective action potential (i.e., the possibility of leading to large-scale protests)
- Could learn how the censors work just by watching and counting

CC BY-SA 3.0 via wallpaperflare.com

King, Gary, Jennifer Pan, and Margaret E. Roberts. 2013. “How Censorship in China Allows Government Criticism but Silences Collective Expression.” *American Political Science Review* 107 (2):326–43. https://doi.org/10.1017/S0003055413000014.

Text analysis

- In its most basic form, text analysis is just about counting words.
- However…
  - counting is easy
  - choosing what to count and why is much harder.

Photo by Mel Poole on Unsplash

Text as data

Photo by Marvin Meyer on Unsplash

Text as data

Photo by Marvin Meyer on Unsplash

Photo by Patrick Tomasso on Unsplash

Text as data

Photo by Marvin Meyer on Unsplash

Analysing text

Photo by Patrick Tomasso on Unsplash

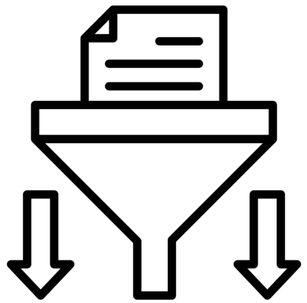
Examples of text as data

What text data can you think of that can be used for CSS research?

Unit of analysis - document

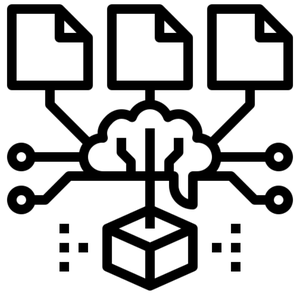
- What is a document? Unit of analysis of text
  - Characterizing spam messages
  - Impact of news on stock market prices
  - Spread of misinformation on online social media platforms
  - Predicting legislator’s partisanship from their floor speeches

Text analysis pipeline



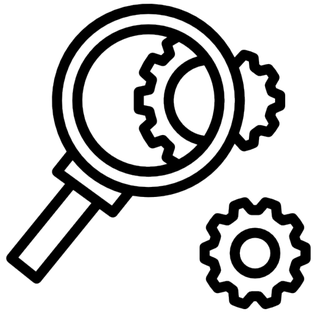
Created by Iconjam - Flaticon

Data preprocessing



Created by Becris - Flaticon

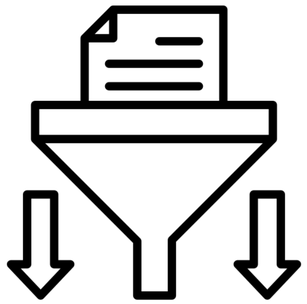
Model building



Created by geotatah - Flaticon

Model evaluation

Representing text as data



Created by Iconjam - Flaticon

Data preprocessing

Reducing complexity

- Retain info useful for automated methods
- Discard info that is likely unhelpful, ancillary, or too complex to use in a model
- Transform text to quantitative data, i.e., go from words to numbers

High dimensionality of text

- A sample of documents,
  - each document has *w* words,
  - each word is drawn from a vocabulary of *p* possible words
  - Unique representation of these documents has p^w dimensions

Photo by Ajda ATZ on Unsplash

High dimensionality of text

- A sample of documents,
  - each document has *w* words,
  - each word is drawn from a vocabulary of *p* possible words
  - Unique representation of these documents has p^w dimensions

- Example, a sample of 60 words tweet, using only 1000 most common words in English language ~ as many dimensions as there are atoms in the universe

Photo by Ajda ATZ on Unsplash

Components of Preprocessing text

- **Tokenization** refers to splitting strings of text into smaller pieces, or “tokens”.
- **Normalization** aims to put all text on a level playing field
- **Noise removal** cleans up the text

Photo by Mick Haupt on Unsplash

Tokenization

- Large chunks of text can be ***segmented*** into paragraphs or sentences

- Sentences can be ***tokenized*** into words

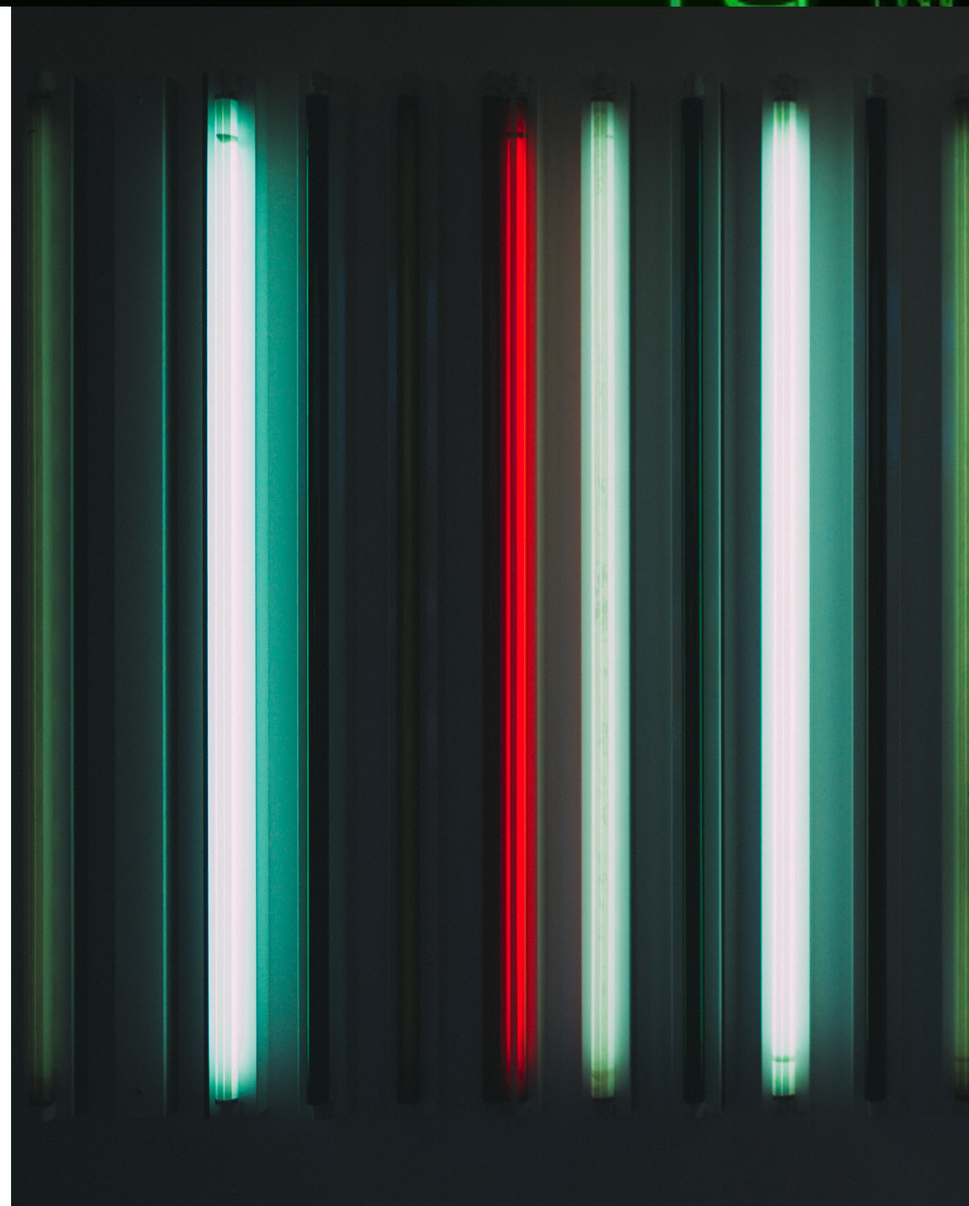


Photo by Ari He on Unsplash

Tokenization

- Large chunks of text can be ***segmented*** into paragraphs or sentences
  - The quick brown fox jumps over the lazy dog. The lazy dog keeps sleeping.

How can we identify sentences in larger text?

Photo by Ari He on Unsplash

Tokenization

- Large chunks of text can be ***segmented*** into paragraphs or sentences
  - The quick brown fox jumps over the lazy dog. The lazy dog keeps sleeping.
  - Dr. Ford did not ask Col. Mustard the name of Mr. Smith's dog.
  - "What is all the fuss about?" asked Mr. Peters.

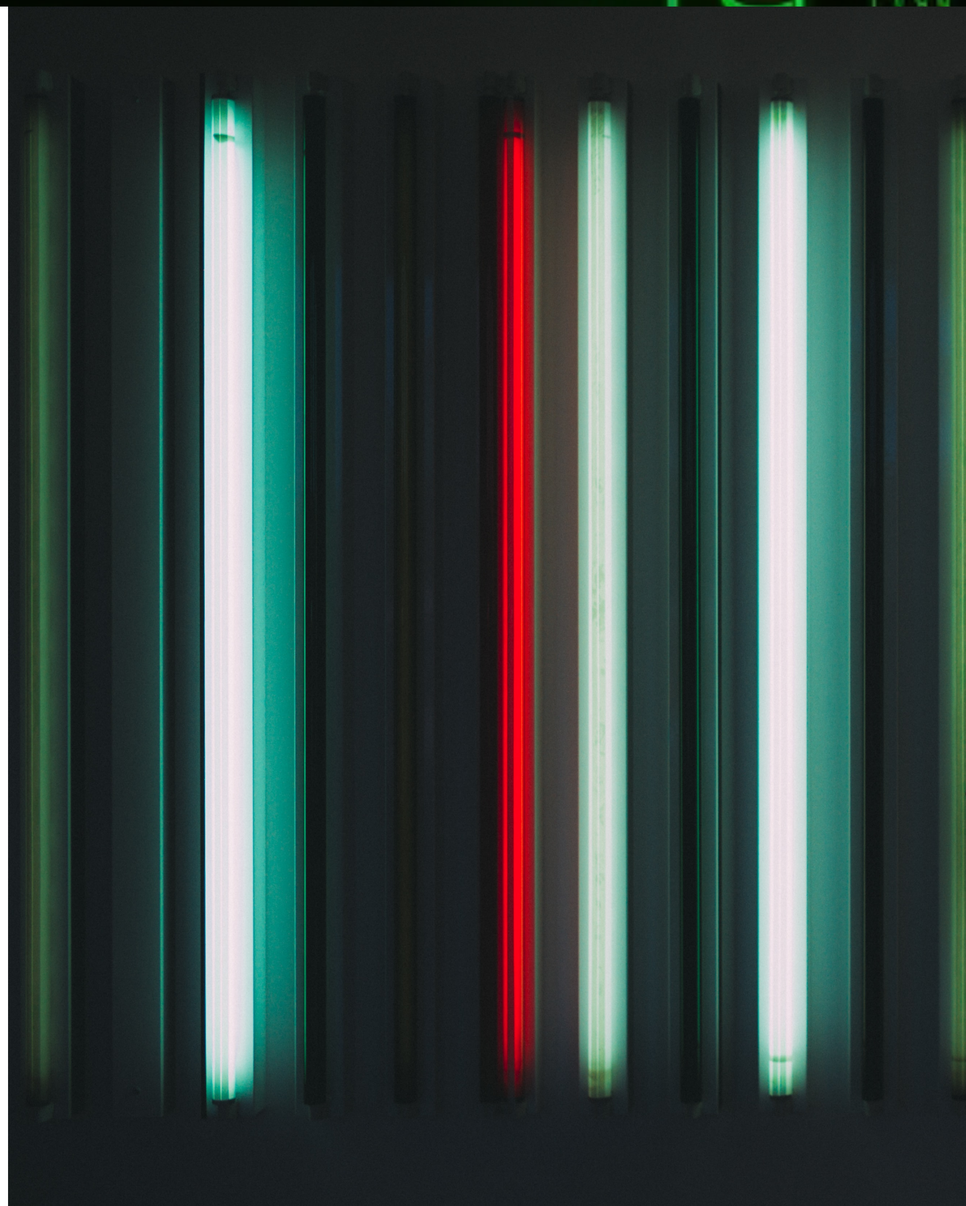


Photo by Ari He on Unsplash

Tokenization

- Large chunks of text can be ***segmented*** into paragraphs or sentences
  - The quick brown fox jumps over the lazy dog. The lazy dog keeps sleeping.
  - Dr. Ford did not ask Col. Mustard the name of Mr. Smith's dog.
  - "What is all the fuss about?" asked Mr. Peters.
- Sentences can be ***tokenized*** into words
  - The quick brown fox jumps over the lazy dog.

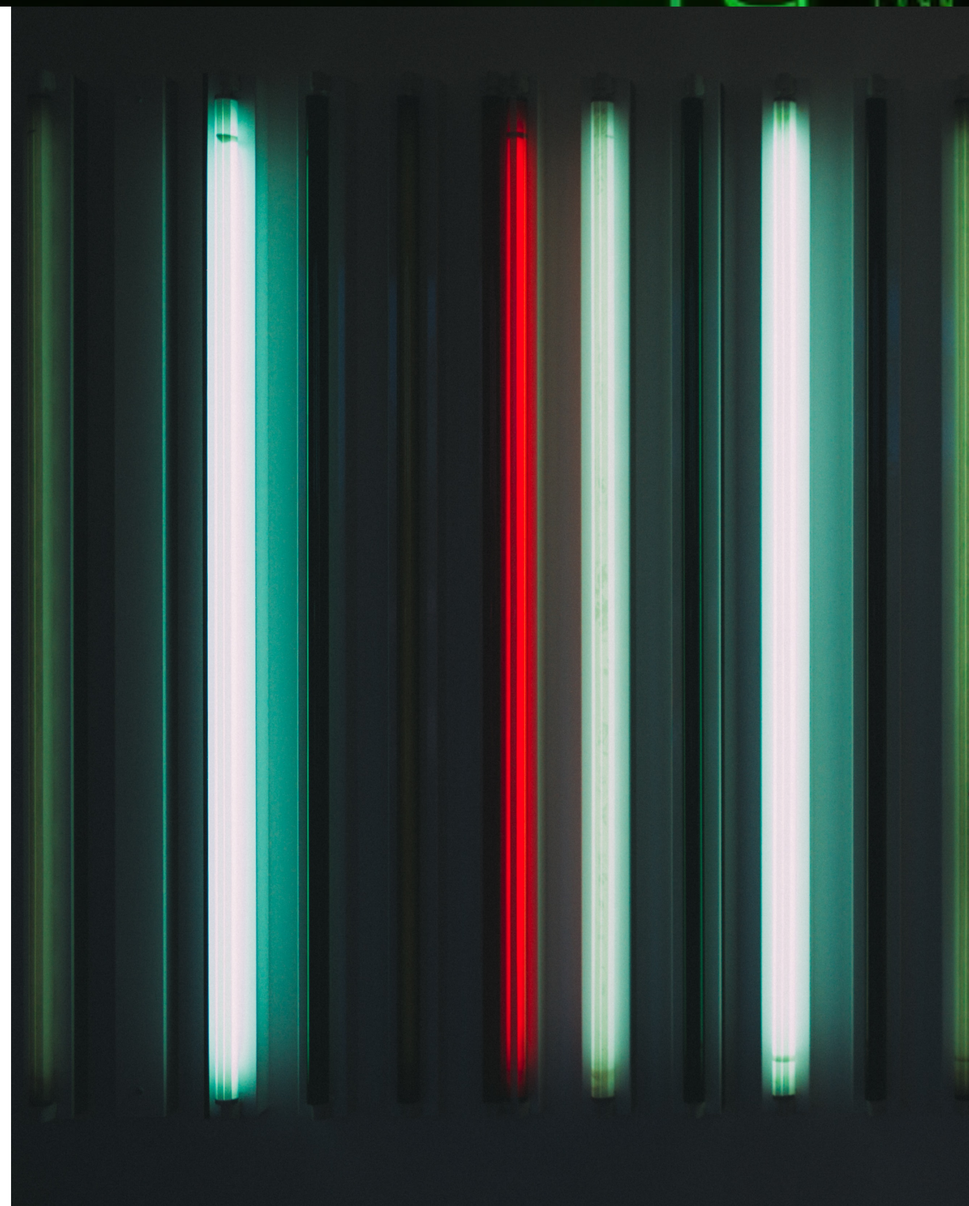


Photo by Ari He on Unsplash

Tokenization

- Large chunks of text can be ***segmented*** into paragraphs or sentences
  - The quick brown fox jumps over the lazy dog. The lazy dog keeps sleeping.
  - Dr. Ford did not ask Col. Mustard the name of Mr. Smith's dog.
  - "What is all the fuss about?" asked Mr. Peters.
- Sentences can be ***tokenized*** into words
  - The quick brown fox jumps over the lazy dog.
  - She’s going to move to New York.

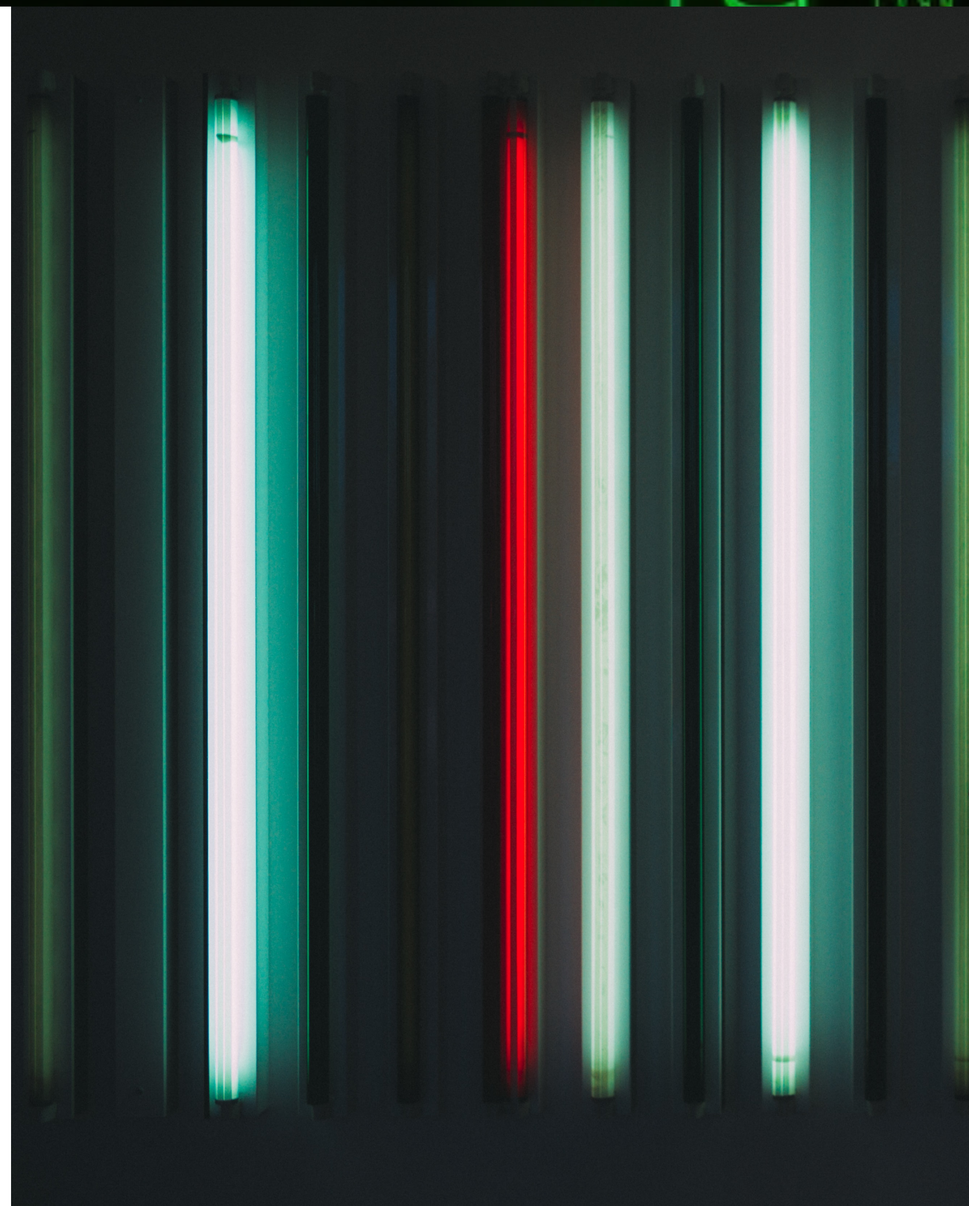


Photo by Ari He on Unsplash

Normalization

- Transforming text into a single canonical form that it might not have had before, puts all text on equal footing for processing to proceed uniformly

Photo by Joshua Sukoff on Unsplash

Normalization

- Transforming text into a single canonical form that it might not have had before, puts all text on equal footing for processing to proceed uniformly

- **Stemming**: Process of eliminating affixes (suffixes, prefixes, infixes, circumflexes) from a word in order to obtain a word stem
  - E.g., running -> run, wait/waiting/waits/waited -> wait

Photo by Joshua Sukoff on Unsplash

Normalization

- Transforming text into a single canonical form that it might not have had before, puts all text on equal footing for processing to proceed uniformly

- **Stemming**: Process of eliminating affixes (suffixes, prefixes, infixes, circumflexes) from a word in order to obtain a word stem
  - E.g., running -> run, wait/waiting/waits/waited -> wait

- **Lemmatization**: capture canonical forms based on a word's lemma
  - E.g., better -> good, ran/runs/running -> run

Photo by Joshua Sukoff on Unsplash

Normalization

- Transforming text into a single canonical form that it might not have had before, puts all text on equal footing for processing to proceed uniformly

- **Stemming**: Process of eliminating affixes (suffixes, prefixes, infixes, circumflexes) from a word in order to obtain a word stem
  - E.g., running -> run, wait/waiting/waits/waited -> wait

- **Lemmatization**: capture canonical forms based on a word's lemma
  - E.g., better -> good, ran/runs/running -> run

- Stemming (nonmeaningful reps, e.g., sentiment analysis) vs Lemmatization (meaningful reps, e.g., chatbots, question answering)

Photo by Joshua Sukoff on Unsplash

Normalization (ctd)

- **Stop word removal**: filter out words that contribute little to overall meaning, as they are the most common words in the language
  - Articles (“the,” “a”), conjunctions (“and,” “or”), forms of the verb “to be” etc.
  - Typically excluded based on a predefined list (generic/custom)
  - E.g., ~~**The**~~ quick brown fox jumps over ~~**the**~~ lazy dog.

Photo by Joshua Sukoff on Unsplash

Normalization (ctd)

- **Stop word removal**: filter out words that contribute little to overall meaning, as they are the most common words in the language
  - Articles (“the,” “a”), conjunctions (“and,” “or”), forms of the verb “to be” etc.
  - Typically excluded based on a predefined list (generic/custom)
  - E.g., ~~**The**~~ quick brown fox jumps over ~~**the**~~ lazy dog.

- **Sparse terms removal**: exclude words that occur rarely
  - Convey meaning, but computational cost for expanding feature set often exceeds their diagnostic value
  - Typically exclude all words that occur fewer than k times for some arbitrary small integer k

Photo by Joshua Sukoff on Unsplash

Normalization (ctd)

- **Stop word removal**: filter out words that contribute little to overall meaning, as they are the most common words in the language
  - Include articles (“the,” “a”), conjunctions (“and,” “or”), forms of the verb “to be” etc.
  - Typically excluded based on a predefined list (generic/custom)
  - E.g., ~~**The**~~ quick brown fox jumps over ~~**the**~~ lazy dog.

- **Sparse terms removal**: exclude words that occur rarely
  - Convey meaning, but computational cost for expanding feature set often exceeds their diagnostic value
  - Typically exclude all words that occur fewer than k times for some arbitrary small integer k

- Misc - lower casing, removing white space/punctuations, remove/convert numbers

Photo by Joshua Sukoff on Unsplash

Noise removal

- More task specific
- Example: Document scraped from the web will be wrapped in HTML or XML tags
  - remove text file headers, footers
  - remove HTML, XML, etc. markup and metadata
  - extract valuable data from other formats, such as JSON

Photo by Jason Leung on Unsplash

Components of Preprocessing text

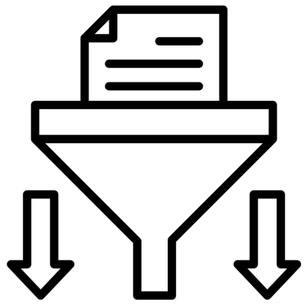
- **Tokenization** refers to splitting strings of text into smaller pieces, or “tokens”.
- **Normalization** aims to put all text on a level playing field
- **Noise removal** cleans up the text

Not a linear process

What is useful info or not depends on the task.
(e.g., Numbers for dates, punctuation for emojis)

Photo by Mick Haupt on Unsplash

Representing text as data



Created by Iconjam - Flaticon

Data preprocessing

Reducing complexity

- Retain info useful for automated methods
- Discard info that is likely unhelpful, ancillary, or too complex to use in a model
- Transform **text to quantitative data**, i.e., go from words to numbers

Feature extraction

- Words -> Numbers (vectors of numbers)
- Text is messy and unstructured (even after the previous preprocessing steps) and models often prefer structured, well defined fixed-length numerical inputs (vectors)
- Feature extraction methods
  - Bag-of-words
  - TF-IDF
  - Word embeddings

Bag-of-words

- Represents a text document as a multiset of its words, disregarding grammar and word order, but keeping the frequency of words.

Photo by Towfiqu barbhuiya on Unsplash

Bag-of-words

- Represents a text document as a multiset of its words, disregarding grammar and word order, but keeping the frequency of words.

- Example,
  - “New York offers new experiences” —> New = 2, York = 1, offers = 1, experiences = 1

Photo by Towfiqu barbhuiya on Unsplash

Bag-of-words

- Represents a text document as a multiset of its words, disregarding grammar and word order, but keeping the frequency of words.

- Example,
  - “New York offers new experiences” —> New = 2, York = 1, offers = 1, experiences = 1

- N-grams - counting phrases (n-grams) instead of words (uni-grams);
  - encodes limited dependence, single words ca be insufficient, e.g., partisan speech analysis “death tax” and “tax break” have strong partisan overtones, lost if we consider unigrams “death”, “tax”, “break” only

Photo by Towfiqu barbhuiya on Unsplash

Bag-of-words

- Represents a text document as a multiset of its words, disregarding grammar and word order, but keeping the frequency of words.

- Example,
  - “New York offers new experiences” —> New = 2, York = 1, offers = 1, experiences = 1

- N-grams - counting phrases (n-grams) instead of words (uni-grams);
  - encodes limited dependence, single words ca be insufficient, e.g., partisan speech analysis “death tax” and “tax break” have strong partisan overtones, lost if we consider unigrams “death”, “tax”, “break” only

- Useful for tasks such as text classification, document similarity, and text clustering

Photo by Towfiqu barbhuiya on Unsplash

Bag-of-words: Limitations

Photo by Towfiqu barbhuiya on Unsplash

Bag-of-words: Limitations

- High dimensionality of feature space may lead to issues of overfitting and computational efficiency
- Lack of context information
- Insensitivity to word associations and semantic relationships between words
- Sparse representations with many elements as zero, increases space and time complexity for computing with them

Photo by Towfiqu barbhuiya on Unsplash

tf-idf

- An approach that excludes both common and rare words
- Very useful in practice

tf-idf

- **Term frequency, tf(t,d)**:
  - measures how frequently a term occurs in a document
  - simplest form - raw count of the number of times the term *t* occurs in document *d*

tf-idf

- **Term frequency, tf(t,d)**:
  - measures how frequently a term occurs in a document
  - simplest form - raw count of the number of times the term *t* occurs in document *d*

- **Document frequency, df(t)**:
  - Number of documents in the collection that contain a term

tf-idf

- **Term frequency, tf(t,d)**:
  - measures how frequently a term occurs in a document
  - simplest form - raw count of the number of times the term *t* occurs in document *d*

- **Document frequency, df(t)**:
  - Number of documents in the collection that contain a term

- **Inverse document frequency, idf(t)**:
  - measure of how much information the word provides, i.e., whether the term is common or rare across all documents
  - idf(t) = log (N/df(t)) where N is the total number of documents

tf-idf

- **Term frequency, tf(t,d)**:
  - measures how frequently a term occurs in a document
  - simplest form - raw count of the number of times the term *t* occurs in document *d*

- **Document frequency, df(t)**:
  - Number of documents in the collection that contain a term

- **Inverse document frequency, idf(t)**:
  - measure of how much information the word provides, i.e., whether the term is common or rare across all documents
  - idf(t) = log (N/df(t)) where N is the total number of documents

- **tf-idf(t,d) = tf(t,d) x idf(t)**

tf-idf

- **Term frequency, tf(t,d)**:
  - measures how frequently a term occurs in a document
  - simplest form - raw count of the number of times the term *t* occurs in document *d*

- **Document frequency, df(t)**:
  - Number of documents in the c

- **Inverse document frequency,**
  - measure of how much informa
    common or rare across all doc
  - idf(t) = log (N/df(t))

- **tf-idf(t,d) = tf(t,d) x idf(t)**

- Highest when term (t) occurs many times within a small number of documents (thus lending high discriminating power to those documents);
- Lower when the term occurs fewer times in a document, or occurs in many documents (thus offering a less pronounced relevance signal);
- Lowest when the term occurs in virtually all documents.

tf-idf: Limitations

- Do not capture any syntactical or semantic information
- High dimensions, can cut off low scored dimensions
- Sparse representation

Word embeddings

- Representation of words as vectors, but efficient and dense
- Distributional hypothesis by Zellig Harris —
  - Words that have similar context will have similar meanings
  - Words in same neighbourhood in the vector space represent similar meaning

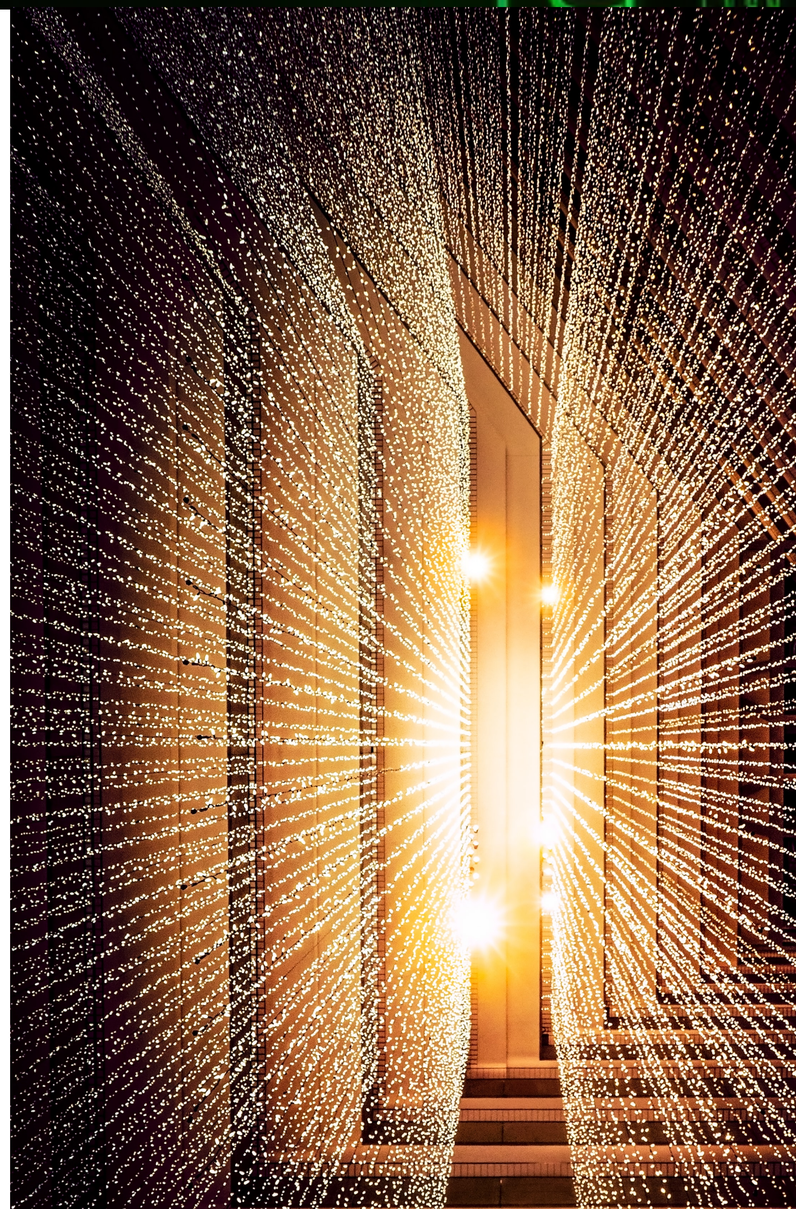


Photo by Joshua Sortino on Unsplash

Word embeddings

- Representation of words as vectors, but efficient and dense
- Distributional hypothesis by Zellig Harris —
  - Words that have similar context will have similar meanings
  - Words in same neighbourhood in the vector space represent similar meaning
- Many methods for creating word embeddings - Word2Vec, Continuous Bag of Words, Skip gram, Glove etc.
- Limitations: memory intensive, corpus dependent => any underlying bias will affect your model

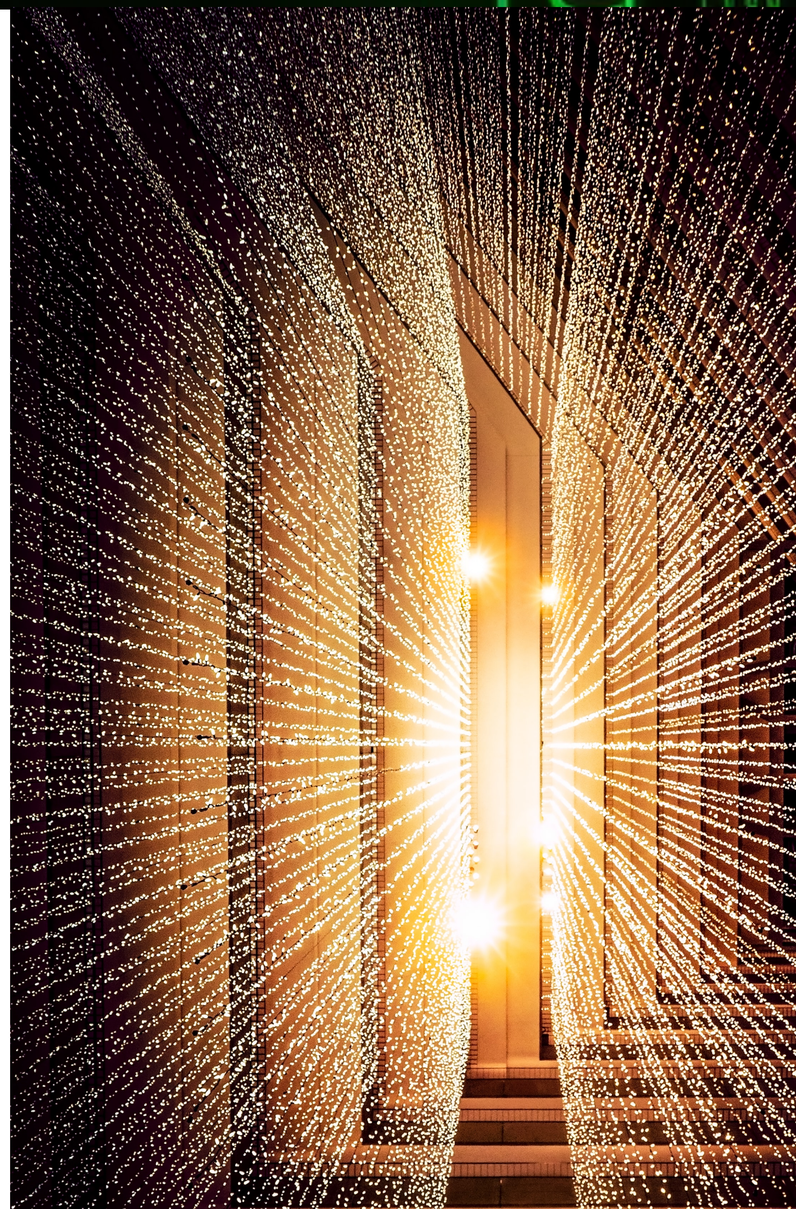
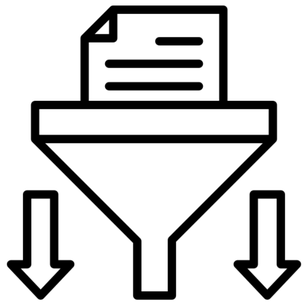


Photo by Joshua Sortino on Unsplash

Representing text as data



Created by Iconjam - Flaticon

Data preprocessing

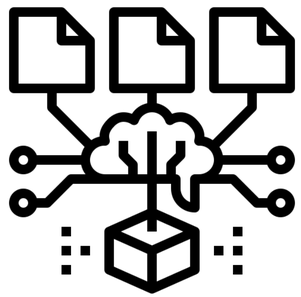
Reducing complexity

- Retain info useful for automated methods
- Discard info that is likely unhelpful, ancillary, or too complex to use in a model
- Transform **text to quantitative data**, i.e., go from words to numbers

Text analysis pipeline

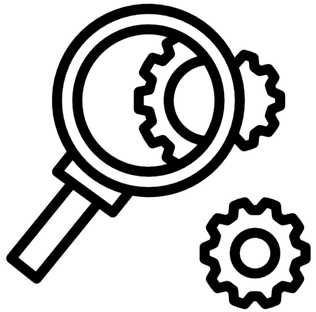
Models and their evaluation

- Different types of models used for analysing text
- Model specific evaluation methods
- End to end evaluation methods



Created by Becris - Flaticon

Model building



Created by geotatah - Flaticon

Model evaluation

Classification models

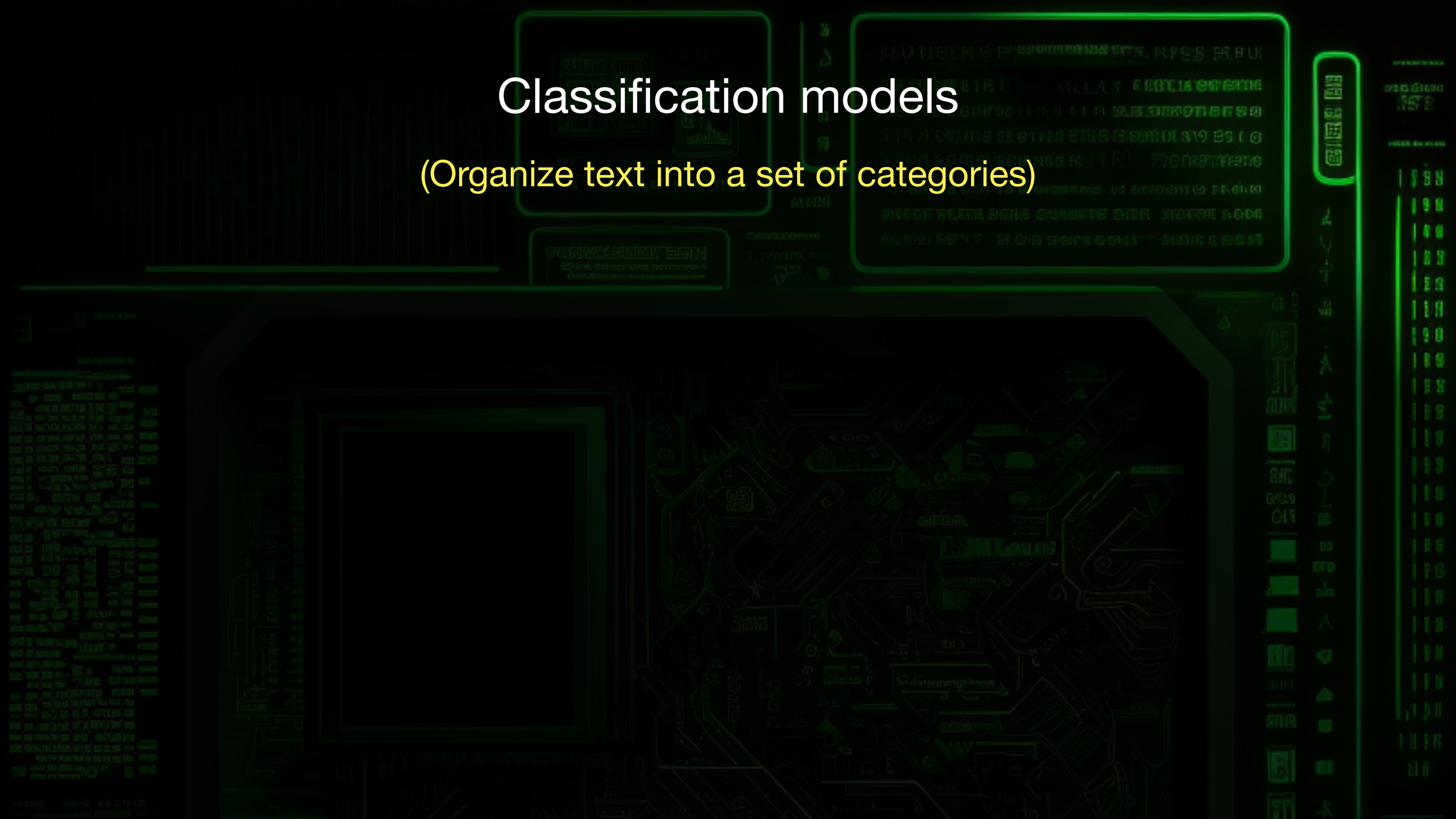
(Organize text into a set of categories)

Expectation management

- Different types of classification models used for CSS research
- Do not discuss details of specific algorithms
- Focus on common characteristics of the algorithms
- For details on specific algorithms, courses on machine learning or deep learning
- Available as easy to use python libraries

Classification models

(Organize text into a set of categories)



Classification models

(Organize text into a set of categories)

Known categories

(minimize the amount of labor needed to classify documents)

Classification models

(Organize text into a set of categories)

Known categories

(minimize the amount of labor needed to classify documents)

Dictionary Methods

Dictionary method

- Most intuitive and easy to apply automated method
- Relies on researcher-defined lists of words (or dictionaries)
- Uses the rate at which keywords appear in a text to classify documents into categories or to measure the extent to which documents belong to particular categories
- E.g., “happy”, “fun” -> positive, “unhappy”, “sad” -> negative sentiments, then “I am *happy* that I had a *fun* vacation, though *sad* that it’s over now” -> positive
- When dictionaries are applied outside the domain for which they were developed, serious errors can occur
- Human validation imperative!

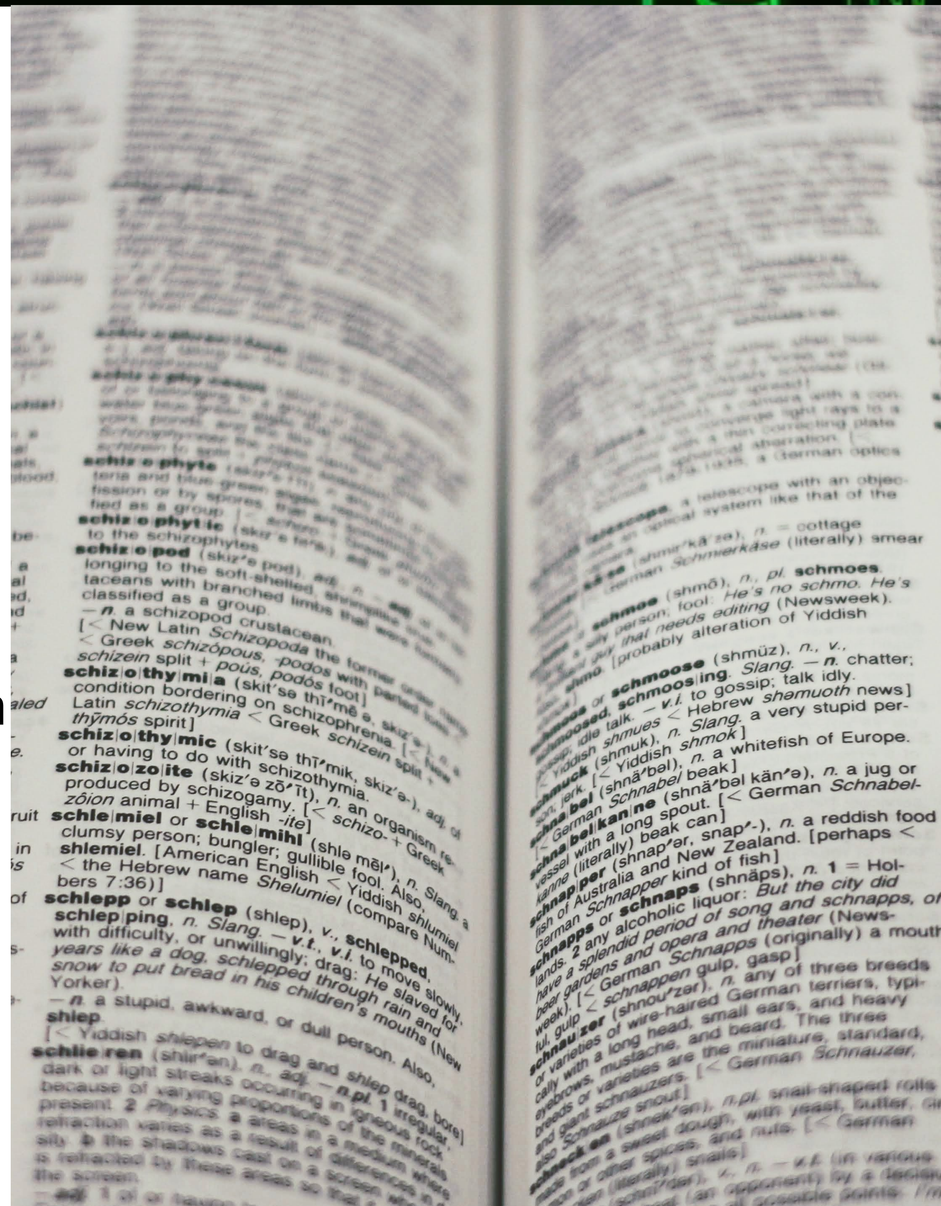


Photo by Joshua Hoehne on Unsplash

Classification models

(Organize text into a set of categories)

Known categories

(minimize the amount of labor needed to classify documents)

- Dictionary Methods
- Supervised Models

Supervised models

- “Learns rules from examples”
- Human coders categorise a set of documents by hand, and algorithms “learns” how to sort the documents into categories using the hand coded training set
- Domain specific, model is trained using a sample of documents from the corpus that is to be classified
- Much easier to validate, with clear statistics that summarise model performance

Photo by Scott Graham on Unsplash

Supervised models (ctd)

- Three basic steps —

- Construct a training set
  - create a codebook (often iteratively) and manually apply the coding scheme to documents
  - Select documents to code (uniform random or stratified)

- Train the model (e.g., Support Vector Machines, Logistic regression, Naive Bayes, Random Forest) using training set and infer categories for the test set

- Validate the model output and classify remaining documents

Photo by Scott Graham on Unsplash

Evaluating supervised models

- If a supervised method is performing well, it will directly replicate the hand coding
- Compare output of machine coding to output of hand coding
- **Split data into training and test set** - avoids overfitting by using out-of-sample data for testing and selecting the model
- **K-fold cross validation**
  - Split training set randomly into k partitions
  - For each of the k partitions, train on all other k-1 partitions and test on the remaining 1 partition to assess the performance

Photo by Scott Graham on Unsplash

Evaluating supervised models (ctd)

Accuracy	The number of correct classifications divided by all classifications performed. Higher values are better.
Confusion matrix	A presentation that plots ground-truth classifications (vertical) against results from the same data's algorithmic classifications (horizontal). In each cell, there are number of cases observed with a particular combination of ground-truth and algorithmic classification. The matrix can be used to ascertain whether there are systematic mistakes (i.e., the classification is wrong for entire categories) or random errors. Having more values along the diagonal is better.
F1 score	An average-based measurement with a contribution from the total number of correct classifications, or true positives, divided by the number of data points actually in the target category (recall) and also from the number of true positives divided by all assignments to that category, including false positives (precision). Higher values are better.

Nelimarkka, M. (2022). *Computational Thinking and Social Science: Combining Programming, Methodologies and Fundamental Concepts*. SAGE.

Classification models

(Organize text into a set of categories)

Known categories

(minimize the amount of labor needed to classify documents)

Dictionary Methods

Supervised Models

Unknown categories

(Discover new ways of organising documents)

Classification models

(Organize text into a set of categories)

Known categories

(minimize the amount of labor needed to classify documents)

Dictionary Methods

Supervised Models

Unknown categories

(Discover new ways of organising documents)

Unsupervised Models

Unsupervised models

- Allowing rules to emerge from the data
- Uses modelling assumptions and properties of texts to estimate a set of categories and simultaneously assign documents to those categories
- Basic process involves conducting many iterations until changes between iterations are sufficiently small — solution has converged
- Allow explorative data analysis without preconceived conceptualisations.
- Always spit out some solution to the problem, need to validate that the algorithm's solution represents the data well —> brings subjectivity to the analysis and is prone to various human biases

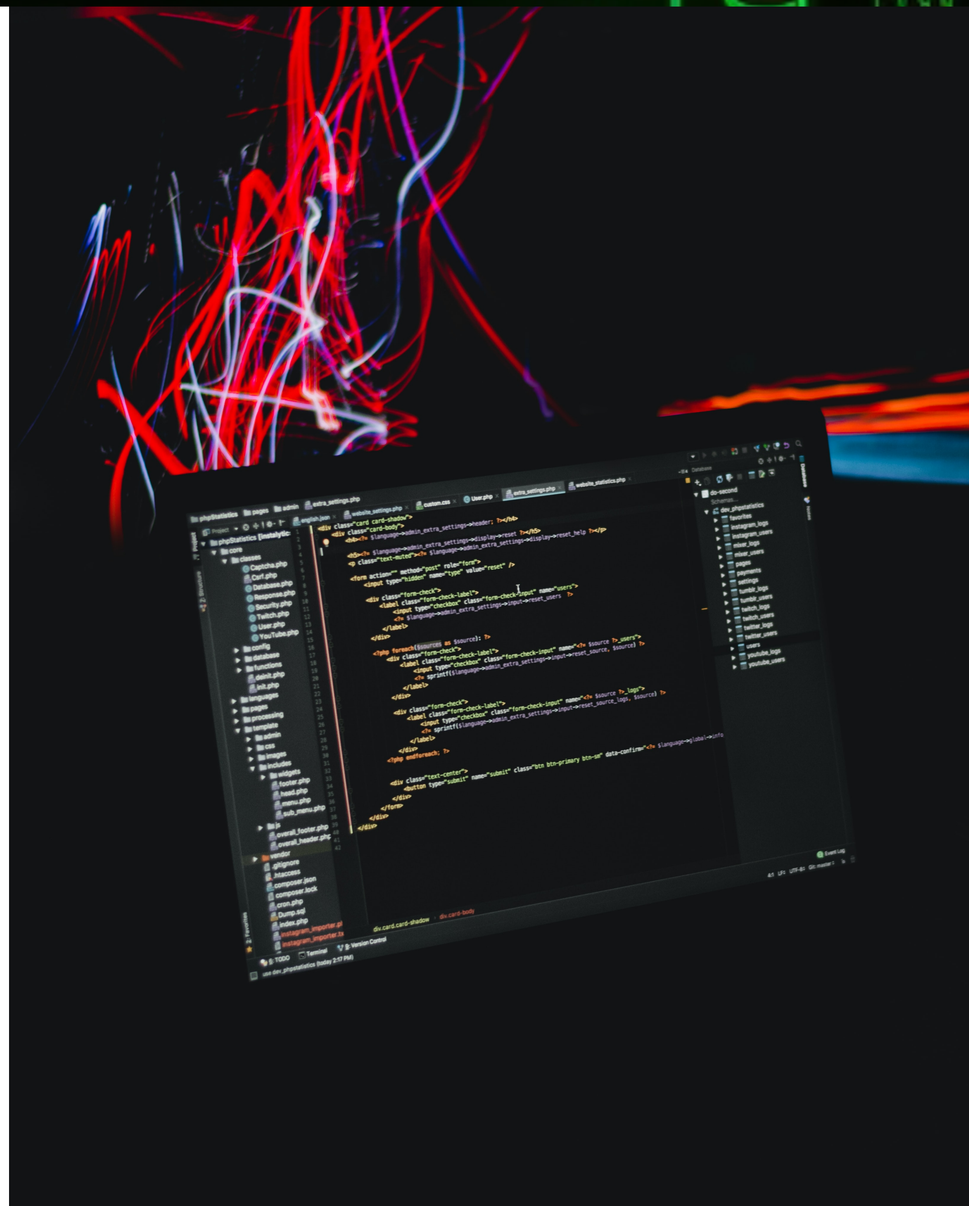


Photo by AltumCode on Unsplash

Unsupervised models (ctd)

- Single membership models (e.g., k-means)
  - Three components - defining documents similarity/ distance, objective function that operationalizes an ideal clustering, optimisation algorithm

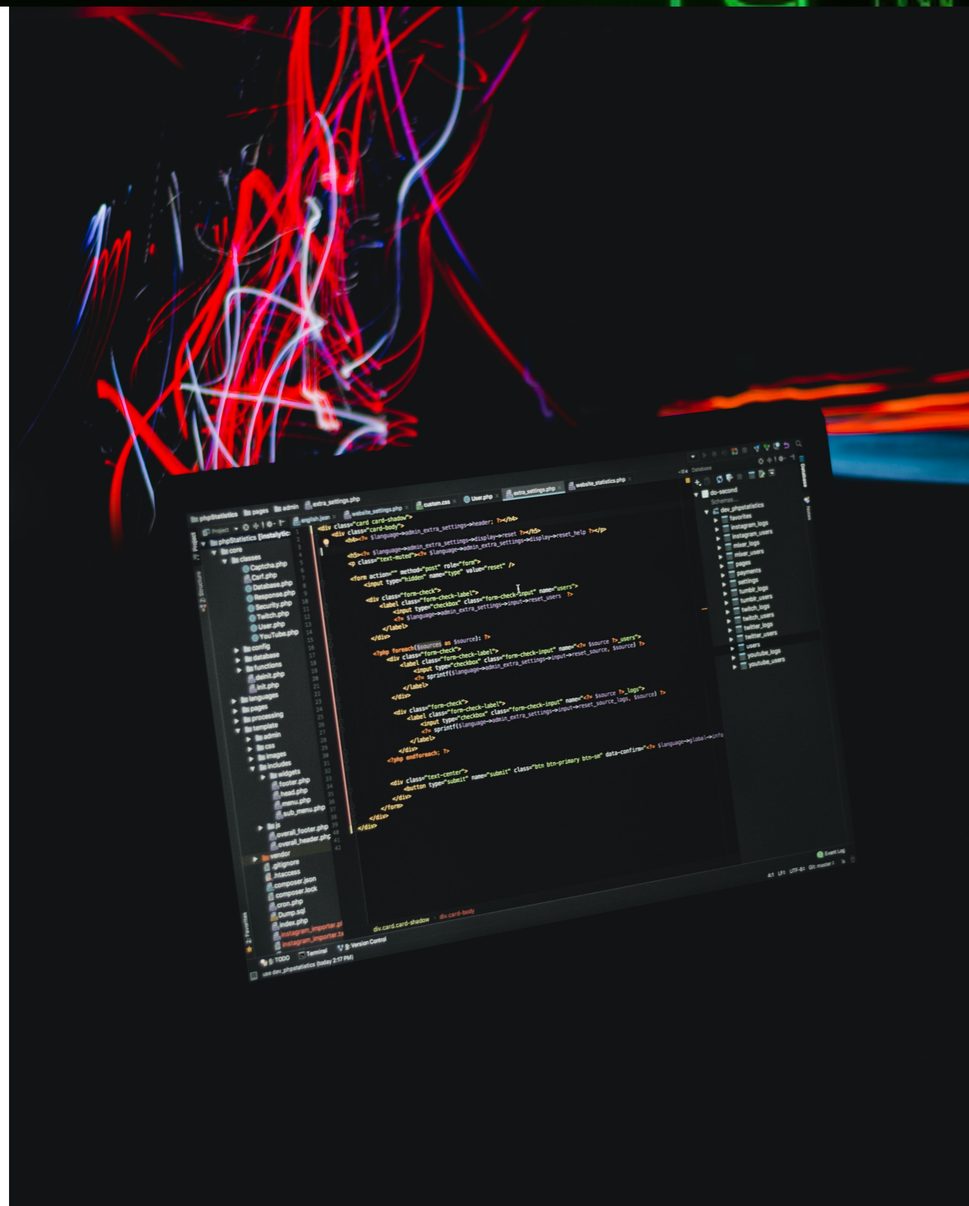


Photo by AltumCode on Unsplash

Unsupervised models (ctd)

- Single membership models (e.g., k-means)
  - Three components - defining documents similarity/ distance, objective function that operationalizes an ideal clustering, optimisation algorithm
- Mixed membership models (e.g., topic models)
  - Two broad characteristics
  - Topic is defined as a probability mass function over words. To estimate a topic, models use co-occurence of words across documents

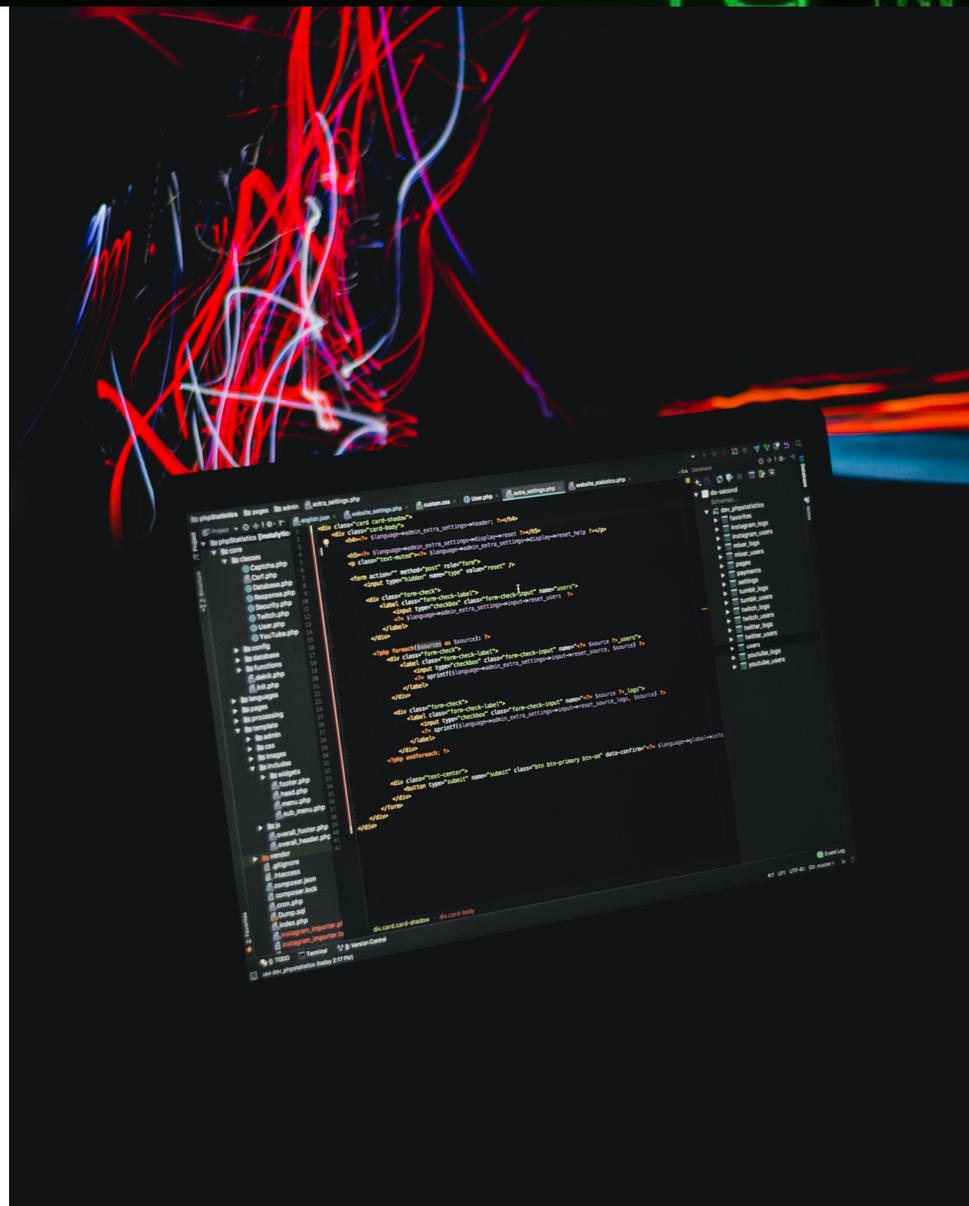


Photo by AltumCode on Unsplash

Evaluating Unsupervised models

- Inspect discriminating words that are highly predictive of documents belonging to a particular topic
- *Semantic validity* - extent to which topics identify coherent groups of documents that are internally homogeneous, yet distinctive from other topics
- *Predictive validity* - how well variation in topic usage corresponds to expected events

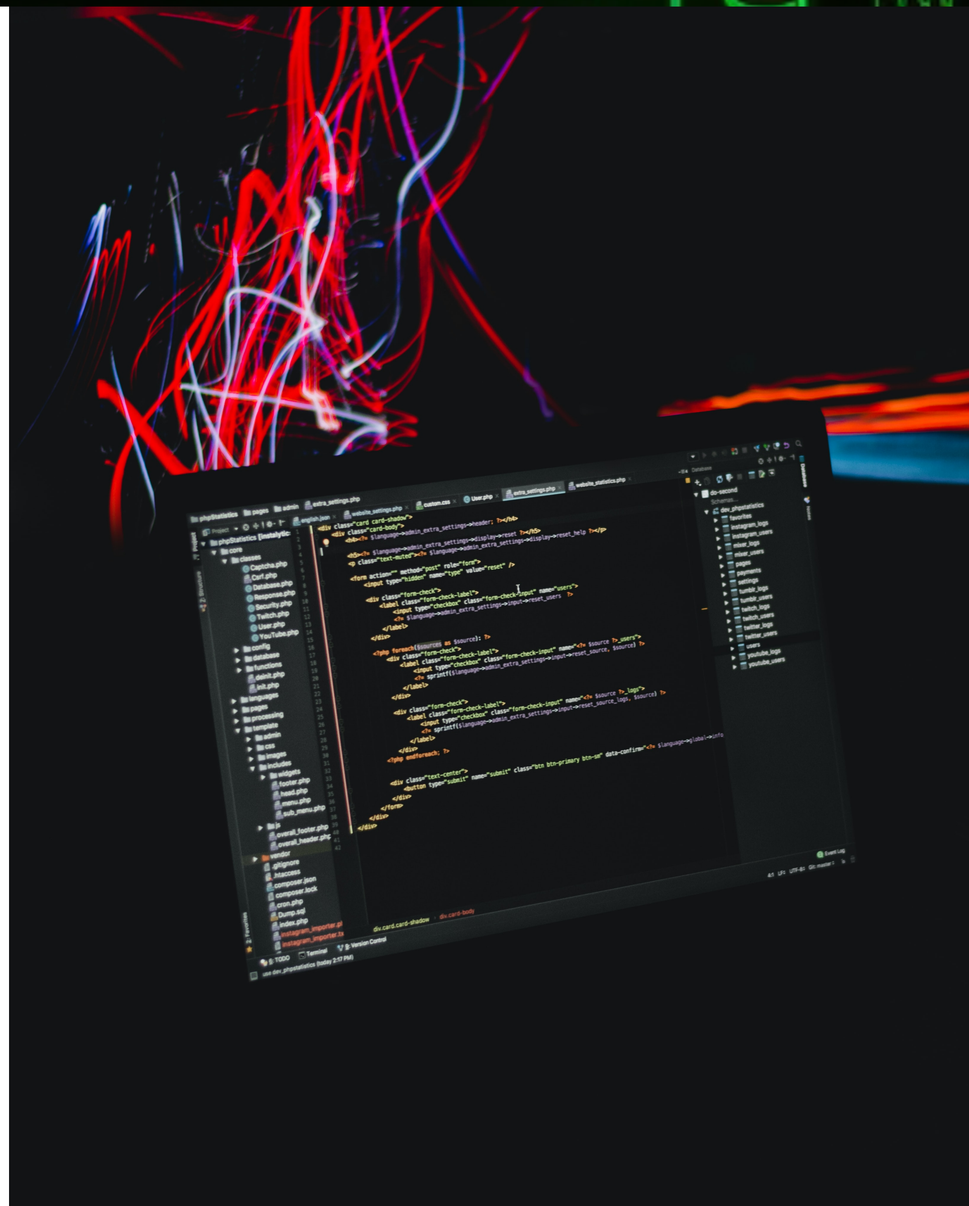


Photo by AltumCode on Unsplash

Classification models

(Organize text into a set of categories)

Known categories

(minimize the amount of labor needed to classify documents)

Dictionary Methods

Supervised Models

Unknown categories

(Discover new ways of organising documents)

Unsupervised Models

*Landscape of models is broader

Caveats of quantitative text analysis

- All quantitative models of language are wrong - but some are useful
- There is no globally best method for automated text analysis
- Validate, validate, validate!

Grimmer, J., & Stewart, B. M. (2013). Text as data: The promise and pitfalls of automatic content analysis methods for political texts. *Political analysis*, *21*(3), 267-297. https://www.jstor.org/stable/pdf/24572662.pdf

Photo by AbsolutVision on Unsplash

References

- Gentzkow, M., Kelly, B., & Taddy, M. (2019). Text as data. *Journal of Economic Literature*, *57*(3), 535-74.All quantitative models of language are wrong - but some are useful

- Grimmer, J., & Stewart, B. M. (2013). Text as data: The promise and pitfalls of automatic content analysis methods for political texts. *Political analysis*, *21*(3), 267-297.There is no globally best method for automated text analysis

- Salganik, M. J. (2019). Bit by bit: Social research in the digital age. Princeton University Press.

- Nelimarkka, M. (2022). Computational Thinking and Social Science: Combining Programming, Methodologies and Fundamental Concepts. SAGE.

- A General Approach to Preprocessing Text Data